9-2018

Event-based allocation of airline check-in counters: A simple dynamic optimization method supported by empirical data

Mahmut PARLAR
McMaster University

Brian RODRIGUES
Singapore Management University, brianr@smu.edu.sg

Sharafali MOOSA
Singapore Management University, sharafalim@smu.edu.sg
DOI: https://doi.org/10.1111/itor.12332

Published in International Transactions in Operational Research, Volume 25, Issue 5, September 2018, Pages 1553-1582
https://doi.org/10.1111/itor.12332

Event-based allocation of airline check-in counters: a simple dynamic optimization method supported by empirical data

Mahmut Parlar[a], Brian Rodrigues[b] and Moosa Sharafali[b]

[a] *DeGroote School of Business, McMaster University, Hamilton, Ontario, L8S 4M4, Canada*
[b] *Lee Kong Chian School of Business, Singapore Management University, 50 Stamford Road, Singapore, 178899, Singapore*
E-mail: parlar@mcmaster.ca [Parlar]; brianr@smu.edu.sg [Rodrigues]; sharafalim@smu.edu.sg [Sharafali]

Abstract

This paper studies the real-life problem of dynamically optimizing the number of airport check-in counters to allocate for a single flight. The main feature of our work is the use of empirical data collected at the Singapore Changi Airport, which drives the dynamic optimization model of a parallel queues system. We propose an event-based dynamic programming model that simplifies considerably the optimization analysis even for large-scale problems with 700+ booked passengers. We investigate the following research questions: (a) For a particular flight, what is the optimal number of counters the system should open with and what is the corresponding optimal total cost? (b) Given the state of the system at any event epoch, should we open another counter or not and what is the optimal cost-to-go from this state? The empirical data we collected at the airport are used to test the assumptions, estimate the key parameters, and run the computational experiments. We apply our model to 14 flights at the Singapore Changi Airport and identify cases in which, depending on the cost parameters, the model advocates the use of either a dynamic or a static policy. Although the model concerns only an exclusive-use system, it is flexible enough to apply to other configurations such as a common-use system or a single-queue, multicounter system.

Keywords: airport operations; queueing; dynamic programming

1. Introduction

The problem studied in this paper was motivated by a study at the Singapore Changi Airport, often voted the "Best Airport in the World."[1] The airport strives to provide better service for its customers, and our focus was to improve queue management at one of its international terminals. Typically, as in many high-volume airports, the passenger check-in service provider must decide the allocation of a limited number of check-in counters to each flight/airline to suit flight schedules and planned

[1] http://www.changiairport.com/en/aboutus.html (accessed 29 January 2016).

passenger loads. The challenge is to complete check-in within a specified time window without having to open additional counters or compromise customer service standards. The practice is to preallocate counters and open additional counters during check-in using experience-based rules of thumb. A major concern for the service provider (a third-party company such as Hong Kong Airport Services Ltd., in Hong Kong; SATS Ltd. and DNATA in Singapore) is to maintain high-quality counter allocation schedules for each day, given variations in skill and experience levels of human schedulers. The effectiveness of the service provider's counter management has high impact on the image of the airport and airlines that operate through it since an efficient schedule translates to shorter queues and faster check-in. In addition, on-time counter closure impacts downstream operations, including baggage processing and flight gate scheduling. It is clear that the service provider has to perform a balancing act in meeting, (a) the demands of the airlines in terms of the number of counters, (b) the service-level standards set by the airport authority, and (c) the concerns about its own bottom line.

Thus, the problem studied in this paper, namely, determining the number of counters to allocate to a single flight with the objective of minimizing counter operating and waiting time costs, is of critical interest to the service provider. The check-in counter system in use is an exclusive-use system dedicated to a single flight. It is worth pointing out that in countries such as the United States, airlines have predominantly moved to using common-use counters that might make our model seem to have limited applicability. But there are many prominent airports around the world, which still use exclusive-use counters. In particular, in Singapore's Changi Airport,[2] some of the terminals employ an exclusive-use counter system. Some years ago, Singapore Airlines decided to do away with self check-in kiosks[3] citing low usage of these kiosks and the growing popularity of Internet and mobile check-ins. Although Changi Airport is now promoting the use of these self check-in kiosks, we are of the view that it will be a very long time for these to become popular, as research evidence points to the fact that "many passengers are reluctant to use new technology within the public sphere for fear of social embarrassment" (Minton, 2008).

On this subject, Parlar and Sharafali (2008) were the first to propose a stochastic dynamic programming (SDP) model based on the analysis of the underlying queue in the transient regime. They modeled the system as a multiserver queue with the arrival process occurring according to a passenger departure process from the finite population of passengers holding confirmed bookings. In queuing parlance, this is referred to as a death process (see Feller, 1968). In their model, the decision to open additional counters was made periodically at equally spaced time instants. This required the time-dependent transition probability functions of the underlying absorbing Markov process. Moreover, the periodic-review dynamic programming approach involved the evaluation of a very large number of these functions that made it difficult to use for solving realistically sized problems. Also, in that work, a single-queue/multiple-counter system was used because for some of the flights, the service provider uses such a system. But at the terminal that we now study in this paper, the service provider used parallel-queue/multiple-counter system due to counter layout and space constraints. Another important observation we made, which was also pointed to us by the service provider, is the fact that queue lengths are almost always equal as a result of jockeying—a

[2]http://www.changiairport.com/en/flight/departures.html (accessed 29 January 2016).

[3]Kaur, K., 2011. SIA pulls plug on kiosks for self check-in. *The Straits Times*, 13 December 2011.

behavior cited and used in the literature (So and Tang, 1996). These and other observations naturally led us to incorporate the following key features in this study:

- *Continuous review*: In this paper, we propose to observe the system continuously with the decision instants being the epochs of arrivals and service completions. Continuous review is consistent with practice since the check-in service provider does monitor the congestion continuously and responds accordingly to open or close counters. In effect, in this paper we employ event-based SDP (Koole, 1998, and other references therein). As the process is terminating, the number of events in the dynamic programming model (i.e., arrival and service completion epochs) is finite, which cannot exceed twice the number of passengers booked for the flight. This facilitates the analysis both analytically and numerically.
- *Parallel queue*: In this paper, the check-in counters system is a multiserver parallel queuing system in which each counter has its own queue. So and Tang (1996) have pointed out that for such systems, it is not necessary to consider the state of the individual queues as the state of the queueing process. This fact adds to the simplification of the analysis as presented here. Indeed, as we will show, ours is the first work to statistically affirm this hypothesis in the "Observation of and hypothesis test for 'perfect' jockeying" section.
- *Use of real data*: The service provider at the airport we visited believes that this problem is very important and facilitated for us in the collection of data on check-in counter operations. This is again the first paper to use realistic data in the analytical model proposed, although simulation-based models have used such data in the past.

Our main contributions in this paper are as follows. We propose and solve a new model that realistically depicts the check-in counter allocation problem faced by the service operator. In fact, we show that the problem is easy to analyze. For example, we find that although there is an underlying basic queue in operation in the system, we do not have to use complicated queueing theory results *per se*. The numerical investigation serves to show that the model is implementable at any terminal, and this can be achieved without difficulty.

The paper also contributes to the literature by way of the statistical hypothesis tests on some features/characteristics/behaviors of the system. For example, we test the hypothesis of "perfect jockeying" by passengers (explained in the "Observation of and hypothesis test for 'perfect' jockeying" section) and also of the functional relationship between service time and group size. Thus, by recording and sharing our experience in the process of real data collection and analysis, the paper also demonstrates the intricacies associated with incorporating real data in analytical modeling studies.

The remainder of this paper is organized as follows. In Section 2, we position our work with the relevant literature. In Section 3, we derive the required basic characteristics of the queueing model. As a prelude to the event-based SDP formulation, we also describe the arrival and service completion processes in this section. Section 4 dwells on the collection and analysis of empirical data. Section 5 develops the SDP model. The data sets are used to justify some of our assumptions and carry out numerical experiments, the results of which are presented in Section 6 together with some insights gained from our analysis. Some other managerial implications are highlighted in Section 7. Suggestions for future research and some concluding remarks on this piece of applied work are given in the last section.

2. Literature review

The problem studied in this paper is of strategic importance to airport operations. Yet, we do not find very many papers in the literature analyzing this issue. Among the scant literature available, one can only find predominantly simulation-based models. To our knowledge, Parlar and Sharafali (2008) was the first work that demonstrated that this problem for *exclusive-use* check-in counter systems[4] is amenable to analytical treatment. In that study (Parlar and Sharafali, 2008), the authors resorted to a periodic-review dynamic programming approach with period-dependent lifetimes. One of the inputs was the time-dependent transition probabilities of the underlying terminating queueing process. The output was the optimal dynamic assignment of counters that minimized a suitable expected cost function. Although the state of the literature has not changed much from what was reviewed in Parlar and Sharafali (2008), for ease of reference, we highlight in this section some of the noteworthy papers. Lee (1966) was the earliest to consider this problem. In that work, the passenger arrival stream was assumed to be Poisson (which is somewhat valid if the system considered is a *common-use* check-in counter system[5]) and an $M/M/s$ queue was used to determine the average wait times, in order to design capacity and space allocation for check-in areas.

Among the simulation-based papers, the work by Chun and Mak (1999) for the Hong Kong airport stands out. The simulation-based decision support system they developed for check-in counter management is indeed comprehensive. They assumed a Poisson arrival stream at the counters and beta-distributed service times to determine the number of counters to open for each flight. A similar work with a different perspective was carried out by Cao et al. (2003) for the Ottawa airport in Canada. They identified the critical factor that impacts the check-in counter performance to be the (passenger) agents' working schedule. The authors developed a linear programming model and a heuristic to determine alternative working schedules for the agents that minimized the total agent-hours, which also met the passenger load that varied throughout the day. Recently, with the objective to improve efficiency of check-in operations, Appelt et al. (2007) used simulation and scenario analysis for the check-in procedure at the Buffalo Niagara International Airport. Almost all the works on the subject employ nonterminating simulation, which is more suitable for common-use check-in counter operations. Such a model is not suitable for exclusive-use operations, as they are dedicated to single flights and the calling population for a single flight is finite. van Dijk and van der Sluis (2006) were the first to use terminating simulation for a dedicated use check-in counter operation.

Recently, Bruno and Genovese (2010) proposed a deterministic static mathematical programming model for check-in counter allocations for a day in Naples International Airport, Italy. Stolletz (2010) considered a hierarchical decision problem of workforce planning for check-in facilities. The input to the problem is the number of counters demanded by the airlines as stipulated in the contract between the service provider and the airlines. The author then used a binary linear programming formulation to identify fortnightly tours for the employees. We highlight that in our work the number of counters is a decision variable and the service provider actually wanted to justify whether the demands placed on it by the airlines were reasonable.

[4]The exclusive-use check-in counter system is a set of counters dedicated to every single flight.

[5]The common-use check-in system consists of a long continuum of counters. Passengers, irrespective of their flight, can check in at any of those open counters.

Hsu et al. (2012) consider a very different problem concerning check-in counter utilization. They assume that all possible modes of check-in are available (i.e., counter check-in, self-service check-in kiosks, online check-in, and barcode check-in) for use by the passengers. The authors' premise is that passengers can be carefully directed to any one of these facilities depending on the state of the congestion. They developed a model to dynamically allocate facilities and passengers with the objective to minimize the waiting time. The authors have validated their model with data from the Taoyuan International Airport, Taiwan. Although the model is very useful, given the observations in the ethnographic study of check-in facility by Minton (2008), it is to be seen whether the passengers will be kind toward such directions from the service provider.

In addition, there is a body of simulation-based research that includes check-in counters as one of the facilities among others in an airport through which the passengers are expected to pass. For example, Fayez et al. (2008) present the methodology and the generic model that will quantify and assess passenger flow in airport terminal functional areas and relate these requirements to the airport's key performance indicators and level of service. For other papers of the same type, we refer the readers to Fayez et al. (2008).

From the foregoing it is clear that check-in counter management is critical to airport check-in service providers and continues to attract the interest of researchers from all over the world. While this is the case, research using mathematical modeling and analysis is scarce in the literature except for Parlar and Sharafali (2008). This is basically due to the perception that this problem is very complex and complicated and so is not amenable to mathematical modeling. This paper attempts to prove that there is an easier way to tackle this problem.

3. Operating characteristics of the basic queue and the model

In this section, we present the basic characteristics of the counter queues that will lead to the formulation of the SDP model. The system in operation is an exclusive-use check-in counter system that is dedicated to a particular flight. The system starts operating with a certain number of counters (say, k), about three hours before the scheduled departure time of the flight and will close its operation about half an hour before the scheduled departure. Thus, the check-in system is open only for a finite duration of T hours to serve that particular flight. The number of passengers booked for the flight is also finite, say N. Arriving passengers choose to join the queue at any open counter. After completing the check-in process at a counter, the passengers leave the check-in system to navigate through other processes before boarding the flight. It is apparent that the underlying queueing system is a multichannel parallel queueing system. From the description above, it is very clear that the system will never reach a steady state.

We make the following assumptions in our modeling, which are motivated by the observations on the ground.

1. Assumption 1 (arrival process): As mentioned above, the calling population for the check-in system is finite. There is a large body of literature on queues with finite population size but unlike queues in this system, the served customers will not return to the population pool. Thus, the system will close once all the passengers have checked in or if the time for closure has been reached, whichever is earlier. Hence, we assume the arrival process to be a "death" process with exponentially distributed lifetimes (see Feller, 1968; Bhat, 1984). In particular, we assume that

the passengers arrive singly to check-in. For details on the estimation procedure related to the arrival process, see Section 4.

2. Assumption 2 (single or parallel queues and behavioral readjustment of the queue length): In the queueing system used in some of the terminals at the airport, each open counter has its own queue. This is a common practice in most of the airports around the world. Some of the reasons advanced for using a parallel queues system instead of adopting a single-queue system are as follows:
   (a) Passengers perceive the waiting time to be longer in a single-queue system than if they joined a shorter queue of a parallel system.
   (b) Passengers also feel that if they wait in a single line, then for the passenger at the head of the queue, the time to search for an idle counter results not only in time wasted, but also creates anxiety.

   Any arriving passenger usually chooses a queue of her choice or is directed by a staff to the queue with the shortest length. Further, the passengers on seeing a shorter queue usually also jockey between queues. Consequently, we assume that the length of all queues is essentially the same at all times. This adjustment is observed in real life, which occurs either naturally, or is directed by a security personnel (see also So and Tang, 1996). In fact, we too observed such a behavior at the Singapore Changi Airport.

   Thus, as pointed out in So and Tang (1996), we do not need to keep track of the number in each queue, and it suffices to track the total number in the system. In Section 4, we will use statistical significance tests with the data collected to justify the assumption of equal queue lengths. We highlight that this phenomenon makes our model applicable to both single-queue or parallel-queue multiserver system.
3. Assumption 3 (state-dependent service rates): We assume that at each counter, there is a provision of a basic service that follows an exponential distribution with mean $1/\mu$. But, the actual service time is dependent on the congestion in front of the counters (and the group size). This is to reflect the observation that the counter agents speed up service when they eyeball longer queues but tend to be slower otherwise. Also, when an agent is idle, she usually assists her neighbor, thus speeding up service. For the regression model that we used to estimate the basic service rate, see Section 4.4.

   Specifically, the instantaneous service rate at any time t at a typical counter j, for a group size of 1 (as per Assumption 3) is

   $$\frac{\mu}{(Q_j+1)^{\gamma}},$$

   where $\gamma \leq 0$ and $Q_j + 1$ is the number in the queue at counter j. Now, if k_b is the number of busy counters, then the instantaneous departure rate from the system is

   $$\sum_{j=1}^{k_b} \frac{\mu}{(Q_j+1)^{\gamma}}.$$

   The above formula requires the state of the queue, that is, $(Q_j + 1)$ at every counter j, thus, the state of the queue(s) for this system is a multidimensional vector. Even though tracking the state of the queue would increase the dimensionality in the analytical model, the perfect jockeying

behavior explained in Assumption 3 implies that the Q_j's are all almost surely equal. Hence, in our model, we track only the total number in the system and not the number at each counter. Thus, to know the number in any counter, we just need to divide the total number in the system by the number of busy servers, which gives the state of the number in front of a typical counter as

$$Q_j + 1 = \frac{\text{No. in the system}}{\text{No. of busy servers}} = \frac{\text{No. in the system}}{k_b}.$$

Further, since we assume that all passengers arrive singly, the instantaneous departure rate at time t from the check-in system is clearly

$$k_b \left(\frac{\text{No. in the system}}{k_b} \right)^{-\gamma} \mu = k_b^{1+\gamma} (\text{No. in the system})^{-\gamma} \mu. \tag{1}$$

Note that, in the above equation, if $\gamma = -1$ then the departure rate becomes independent of the number of busy servers (or the number of counters), and if $\gamma = 0$ then the departure rate does not depend on the congestion.

4. Assumption 4 (decision to add a new counter):
   (a) At the decision epochs (i.e., an arrival or a service completion), we open at most only one counter. This assumption can be justified by observing that only one person may arrive at any event time.
   (b) No counter is closed once check-in for the flight is started. This is due to the behavioral readjustment mentioned above, as once a counter is opened, it will be kept busy most of the time until all counters close at time T. This assumption is supported by the empirical data collected as shown in Section 4.
   (c) The above assumption may, however, lead to a situation in which the number of passengers waiting is far less than the number of counters open. From the operator's point of view, this is not favorable and so we use appropriate costs to discourage opening of additional counters when smaller number of passengers are present or expected.
5. Assumption 5 (arrivals and service completions finished by time T): The implication of this assumption is that all passengers arrive before T to be cleared through the check-in process by time T. At the outset, it may seem that there is no guarantee that this will happen. But a little reflection on what airlines currently practice will confirm that almost surely this assumption is valid. Many airlines encourage passengers to book online by making it a bit cheaper to do so. International Air Transport Association (IATA) claims, "On 1 June 2008, the industry moved to 100% electronic ticketing."[6] Booking online involves a commitment to show up as the customers are made to pay upfront with conditions of penalties for cancellations or change of itinerary. So, an online customer is committed to the travel and our premise is that "no-shows" are rare in the new economy, especially for international travel. Even if a particular airline faces no-shows frequently, the common practice is that an airline would use "overbooking" to offset this. Hence, it is more likely that all passengers would arrive by time T.

Table 1 lists, in alphabetical order, the notation we use with a brief description of each symbol.

[6]http://www.iata.org/whatwedo/stb/Pages/e-ticketing.aspx (link active on 12 June 2015.)

Table 1
List of notation in alphabetical order

Symbol	Description
$A(t) = a$	[**state**] The number of passengers who have arrived by time t; $A_n \equiv A(T_n)$
b	The number left behind in the queue when a marked customer leaves
β_0	Fixed cost for opening a check-in counter (\$/counter)
β_1	Penalty cost for an idle counter (\$/counter)
c	[**decision**] The number of check-in counters to open
C_s	The unit variable cost of operating a counter (\$/counter-time)
C_w	The unit cost of making a passenger wait (\$/passenger-time)
$G(c; a, s, k)$	Cost that accrues due to opening and operating the open counters between any two consecutive event epochs
$L(c; a, s, k)$	$= W(c; a, s, k) + G(c; a, s, k)$
$1/\lambda$	Mean "lifetime" of a passenger (time/passenger/time)
μ	The base service rate when only one counter is open and only one customer is in the system (passenger/time)
K	Maximum number of counters available for a flight
k	[**state**] The number of counters already open; with k_n as the number of counters open at T_n
N	The number of passengers booked for the flight
$p_{a+1}(c; a, s, k)$	Conditional probability that the next event at T_{n+1} is an arrival given the current state (a, s, k) at T_n and c new counters are opened
$p_{s+1}(c; a, s, k)$	Conditional probability that the next event at T_{n+1} is a service completion given the current state (a, s, k) at T_n and c new counters are opened
T	Duration of time the check-in counter system will be opened
T_n	The instant of the nth arrival or service completion event
$S(t) = s$	[**state**] The number of passengers who have been served by time t; $S_n = S(T_n)$
$V_n(a, s, k)$	Value function at the current state (a, s, k) at T_n
$W(c; a, s, k)$	Waiting cost that accrues between any two consecutive event epochs

3.1. The queueing process

We recall that the queueing model in operation is a queue with finite population with a difference that the system will close at T. Our objective is to find the optimal counter opening policy. To facilitate this, we consider the stochastic process $\{(A(t), S(t), K(t)), t \geq 0\}$, where $A(t)$ is the number of passengers who have arrived by time t, $S(t)$ is the number of passengers cleared check-in by time t, and $K(t)$ is the number of counters open at time t, respectively. We note that $A(t) - S(t)$ gives the number of passengers still waiting in the system at time t.

3.2. The transition probabilities

From our assumptions on the underlying probability distributions, it is now easy to see that the embedded process $\{(A_n, S_n, K_n), n \geq 0\}$, embedded at T_n, the instant at which the nth event occurs, is a Markov chain. To calculate the transition probabilities, at the occurrence of the nth event at epoch T_n we observe, (a) the number of arrivals (a_n), (b) the number of service completions (s_n), and (c) the number of counters that are already open (k_n). Thus, the state vector at the occurrence of

the nth event is (a_n, s_n, k_n); we denote this as (a, s, k), where no confusion arises. On observing the state (a, s, k) at T_n, we decide either (a) not to open any new counters ($c_n = 0$) or (b) open one new counter ($c_n = 1$). This is indeed reasonable as at any event epoch, the change in the number in the system will be either an increase or a decrease by 1 only. Thus, for a given policy $c_n = c_n(a_n, s_n, k_n)$, the state of the system evolves according to the dynamics,

$$(a_{n+1}, s_{n+1}, k_{n+1}) = \begin{cases} (a_n + 1, s_{n+1}, k_n + c_n), & \text{if } T_{n+1} \text{ is an arrival epoch} \\ (a_n, s_n + 1, k_n + c_n), & \text{if } T_{n+1} \text{ is a service completion epoch,} \end{cases}$$

where $a \geq s$ since the total number of arrivals cannot be less than the total number of service completions.

Our problem is to determine the optimal counter opening policy c_n that will minimize a suitable expected cost function. For this purpose, we formulate and solve an SDP problem that requires expressions for the transition probabilities between the states.

3.2.1. Probability that next event is an arrival

Let us denote by $p_{a+1}(c; a, s, k)$ the probability that the next event is an arrival given that the current state is (a, s, k) and we decide to open c new counters, where $c = 0, 1$. The following proposition provides an explicit expression for this transition probability.

Proposition 1. *When we decide to open $c = 0, 1$ counters after observing the state vector (a, s, k), the probability that the next event is an arrival is given as*

$$\begin{aligned} p_{a+1}(c; a, s, k) &= \Pr\{(A_{n+1}, S_{n+1}, k_{n+1}) = (a + 1, s, k + c) \mid (A_n, S_n, k_n) = (a, s, k)\} \\ &= \frac{(N - a)\lambda}{(N - a)\lambda + [\min(a - s, k + c)]^{1+\gamma} (a - s)^{-\gamma} \mu}. \end{aligned} \tag{2}$$

Proof. Note that when we observe (a, s, k), we still have $(N - a)$ passengers who have not arrived and so the time until the arrival of the next passenger is exponential with rate $(N - a)\lambda$. Further, since $(a - s)$ passengers are still in the system with $(k + c)$ counters open and operating, as per our Assumption 3, the number of busy servers will be $\min(a - s, k + c)$. So, the time until the departure of the next passenger in the system is also exponential with rate $[\min(a - s, k + c)]^{1+\gamma} (a - s)^{-\gamma} \mu$, as given in (1). Thus, using the properties of competing exponentials, the result follows. This proves the proposition. □

It is easy to see that when all passengers have arrived, that is, when $a = N$, the next event cannot be an arrival, so $p_{a+1}(c; N, s, k) = 0$. Similarly, when $a < N$ and all arrived passengers have already been served, that is, when $a = s$, the next event must then be an arrival, that is $p_{a+1}(c; s, s, k) = 1$, as expected. For all other feasible combinations of (a, s), the probability $p_{a+1}(c; a, s, k)$ assumes a value between 0 and 1.

3.2.2. Probability that next event is a service completion

Let us now denote by $p_{s+1}(c; a, s, k)$ the probability that the next event is a service completion given that current state is (a, s, k) and we decide to open c new counters, where $c = 0, 1$. The following proposition provides an explicit expression for this transition probability:

Proposition 2. *When we decide to open $c = 0, 1$ counters after observing the state vector (a, s, k), the probability that the next event is a service completion is given as*

$$p_{s+1}(c; a, s, k) = \frac{[\min(a - s, k + c)]^{1+\gamma} (a - s)^{-\gamma} \mu}{(N - a)\lambda + [\min(a - s, k + c)]^{1+\gamma} (a - s)^{-\gamma} \mu}, \tag{3}$$

provided that $a \geq s + 1$.

Proof. The proof follows by observing that $p_{s+1}(c; a, s, k) = 1 - p_{a+1}(c; a, s, k)$ because the next event must be either an arrival or a service completion. □

4. Collection and analysis of empirical data

In this section, we describe the collection and analysis of the empirical data to test the assumptions and estimate the parameters of the relevant processes, such as the lifetime parameter and service rate. In order to achieve this, we teamed up with the airport authority and service provider to employ an experienced third party to collect the data.

We recall that in our problem, all underlying processes are transient. Keeping this in mind, we designed the data collection exercise. Due to time and budget constraints, we collected data over a period of one week (seven days) on two randomly chosen flights per day. In consultation with the Singapore Changi Airport authority and surveyors, we chose this one week to be a week with normal conditions of operations so as to avoid unusual check-in conditions. We note here that the current work explicitly assumes that the normal conditions prevail. This implies that abnormal conditions such as surges and spikes of arrivals occurring especially toward the closure of counter operations do not happen. Our data also support this, as we do not discern any surges of arrivals. Such abnormal conditions can be handled by the model considered in Parlar and Sharafali (2008).

Having chosen a normal week for data collection, we then randomly selected two flights for each day. The distribution of the 14 randomly selected flights (with seven different international destinations) consisted of six morning flights (numbered F01–F06), four midday flights (numbered F07–F10), and four evening flights (numbered F11–F14) listed in Table 2. A pilot run of this exercise was conducted from which we decided to use six to eight enumerators who were well trained in data collection. The data collected were on (a) the status of the counter queues and behavior of passengers, (b) the arrival process, and (c) the service process. In what follows, we provide descriptive statistics, results of statistical tests and estimation procedures pertaining to the operation of counters, and the analysis of arrival and service processes.

4.1. Operation of counters

The duration of time the counters were open ranged from 1 hour 8 minutes (F09) to 2 hours 45 minutes (F14) and the average time a counter was open was 1 hour 34 minutes. The number of

Table 2
Details of the 14 randomly selected flights with destinations, the number of passengers booked, the actual time period (and the length of time) the counters were kept open, and the total number of counters assigned

Flight	Destination	N	Time period the counter(s) open (24-hour clock)	Duration (hours:minutes)	Number of counters used
F01	Kuala Lumpur	81	03:59–05:28	1:29	1
F02	Hanoi	156	04:00–05:20	1:20	3
F03	Haikou	121	03:48–05:58	2:10	2
F04	Bangkok	131	05:06–06:28	1:22	2
F05	Manila	111	08:35–10:00	1:25	4
F06	Macau	137	10:35–11:50	1:15	3
F07	Kuala Lumpur	145	10:57–12:22	1:25	2
F08	Perth	185	12:55–15:15	2:20	2
F09	Manila-Clark	151	14:55–16:03	1:08	4
F10	Bangkok	167	15:50–17:15	1:25	3
F11	Macau	131	16:45–18:05	1:20	2
F12	Kuala Lumpur	138	18:45–20:05	1:20	3
F13	Chennai	261	18:58–20:32	1:34	4
F14	Manila	172	21:25–00:10	2:45	2

passengers per flight, N, varied from 81 to 261 with an average of 149 passengers per flight. The maximum number of counters used is four and the minimum is one. For other details including the flight destination, time period the counters were open, and the number of counters used, see Table 2.

4.1.1. Observed policy on the number of open counters

To help our modeling efforts, we wanted to know whether any open counters are closed prior to time T and, if so, the time when this happens. To ascertain this, an enumerator periodically observed (every five minutes) the number of counters operating; Table 3 records the findings. The table reveals that in the early stages, counters are never closed. However, as the check-in system's closure time approaches and if at that time only a few unchecked passengers remain, decisions are made to close some counters. The number of counters closed during the period is also not significant. (Indeed, we were informed by the service provider that even if an open counter is closed, fees for the counter are already accounted for, resulting in less incentive to close counters early.)

4.1.2. Observation of and hypothesis test for "perfect" jockeying

In Section 3, we stated that the check-in system has independent queues before every counter open. We also highlighted that the consequence of having multiple queues is that the passengers jockey either spontaneously, or due to the directions given by the check-in/security personnel. As a result of this "perfect" jockeying, the lengths of all queues remain essentially the same at all times, although different at different points in time. This phenomenon is also observed in other systems; for example, So and Tang (1996) have observed this type of jockeying in supermarket checkout counters. Thus, as pointed out in So and Tang (1996), it suffices for us to keep track of only the total number in the

Table 3
Number of counters open observed at five-minute intervals

Flight	Number of counters available from opening to closing
F01	1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0
F02	2,3,3,3,3,3,3,3,3,3,3,3,3,3,3,1,1,0
F03	2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1,1,1,0
F04	1,1,2,2,2,2,2,2,2,2,2,2,2,2,1,0
F05	4,4,4,4,4,4,4,4,3,3,3,3,3,3,3,3,1,0
F06	2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,0
F07	2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1,0
F08	2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,0
F09	3,3,3,3,3,4,4,4,4,4,4,4,4,3,2,0
F10	2,2,2,2,3,3,3,3,3,3,3,3,3,3,3,2,2,0
F11	1,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,0
F12	2,2,2,2,3,3,3,3,3,3,3,3,3,3,3,3,1,1,0
F13	3,4,4,4,4,4,4,4,4,4,4,4,4,4,4,3,1,1,0
F14	2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,1,0

system instead of the number in each queue. It is easy to see that this observation makes our model applicable to single queue, and also multicounter systems.

Even though researchers such as So and Tang (1996) have assumed perfect jockeying in their work, to our knowledge, this assumption has never been tested statistically. In order to justify this assumption in our model and to test it, we assigned enumerators to each counter, who observed the queue sizes every five minutes for all counters that were open for all 14 flights.

With the collected data, we first plotted simple graphs of the queue sizes for each flight. We found that for most of the flights, the queue sizes over time were indeed quite close to each other; see, for example, Fig. 1 for the plot of queue sizes for flight F08 to Perth. In order to statistically test this hypothesis, we used two nonparametric tests: (a) The sign test if the flight used only two counters and (b) Kruskall–Wallis H test if the flight used three or more counters (for details on these tests, see Aczel and Sounderpandian, 2009). In the case of three or more counters, we applied the test on the differences of the queue sizes between counters 1 and 2, counters 2 and 3, etc. For all the flights, the data supported the null hypothesis that the queue sizes were all statistically equal. (For example, for the Perth flight F08, we found that the p-value for the sign test was equal to 0.67; a substantial value that makes it impossible to reject the null hypothesis for any reasonable value of the Type I error.) We emphasize again that this is an important assumption that helps to consider the total number of passengers in the system instead of the vector of individual queue sizes as the state of the process.

4.2. Arrival profile and the estimation of the lifetime parameter λ

The enumerators assigned for this exercise tracked every arrival after the system opened, and noted down (a) the instant of their arrival, (b) the counter they joined, and (c) the group size.

To understand the arrival profile, we define the "lifetime" of a passenger to be the difference between the time of arrival and the time of opening of the counter (and note that this time is not the traditional interarrival time). For example, if the counters open at 3:00 p.m., and a passenger

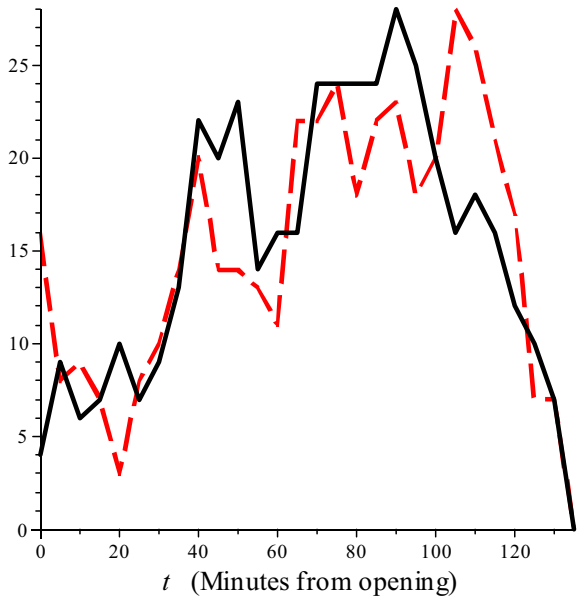

Fig. 1. Counter queues for flight F08 to Perth. For this flight, two counters (#11 and #13) were opened at 12:55 ($t = 0$ minutes) and closed at 15:15 ($t = 135$ minutes). The number of passengers in front of each counter was sampled at five-minute intervals; counter #11 numbers are indicated by dashed line and counter #13 by solid line.

arrives at 3:15 p.m., her "lifetime" would be 15 minutes long. Columns (2) and (3) of Table 4 show the number of passengers booked (or checked-in at the counters), N, and the average lifetimes of these passengers for the 14 flights (which varied from a minimum of 18 minutes 9 seconds for F05 seconds to a maximum of 1 hour 5 minutes 45 seconds for F14).

It is interesting to note that for similar capacity flights, different number of counters have been employed. For example, for F02 (Hanoi), a flight with 156 checked-in passengers, three counters have been used with the average queuing time of 16:28 minutes, and for F09 (Manila-Clark) with 151 passengers, four counters have been used with the average queuing time of 05:09 minutes; see Table 4. These observations seem to suggest that the decision on the number of counters to open is done more on an *ad hoc* basis. In fact, the arrival profiles we have obtained for these two flights reveal that a higher percentage of customers have arrived early for flight F09 and so the service provider might have responded with four counters.

Using the arrival data, we have analyzed the lifetime distributions of all combined flights as shown in Fig. 2. This figure shows that there is a peak around 0 and for higher values of lifetimes, the percentage of observations reduce indicating that most passengers do arrive within a short time of the counter opening. This observation implies that the lifetime distributions can be reasonably approximated as an exponential. When we analyzed the specific flights, we found that except for flights F01, F03, and F12, the profiles for the other flights revealed an approximately exponentially distributed lifetimes.

Next, we tested whether the population of passengers is homogenous across all flights in terms of their mean lifetimes. A one-way ANOVA of the lifetimes for all the flights and the pairwise analysis

Table 4
The average lifetime of all passengers in 14 flights and the average time spent in queue and service for selected samples of marked customers

Flight	*N*	Average lifetime (hours:minutes:seconds)	Sample size	Average time in queue (hours:minutes:seconds)	Average time in service (hours:minutes:seconds)
F01	81	0:47:46	45	0:06:08	0:01:14
F02	156	0:26:23	51	0:16:28	0:02:18
F03	121	0:59:48	20	0:05:51	0:01:51
F04	131	0:24:05	60	0:12:31	0:01:15
F05	172	0:18:09	51	0:09:19	0:02:40
F06	137	0:23:05	37	0:05:11	0:01:23
F07	145	0:34:59	35	0:01:30	0:02:49
F08	185	0:46:33	38	0:01:47	0:00:59
F09	151	0:18:34	44	0:05:09	0:01:51
F10	167	0:38:22	35	0:08:54	0:01:22
F11	131	0:24:49	39	0:06:21	0:01:08
F12	138	0:39:14	42	0:01:06	0:01:13
F13	261	0:25:24	32	0:12:03	0:02:09
F14	111	1:05:45	62	0:00:57	0:01:21

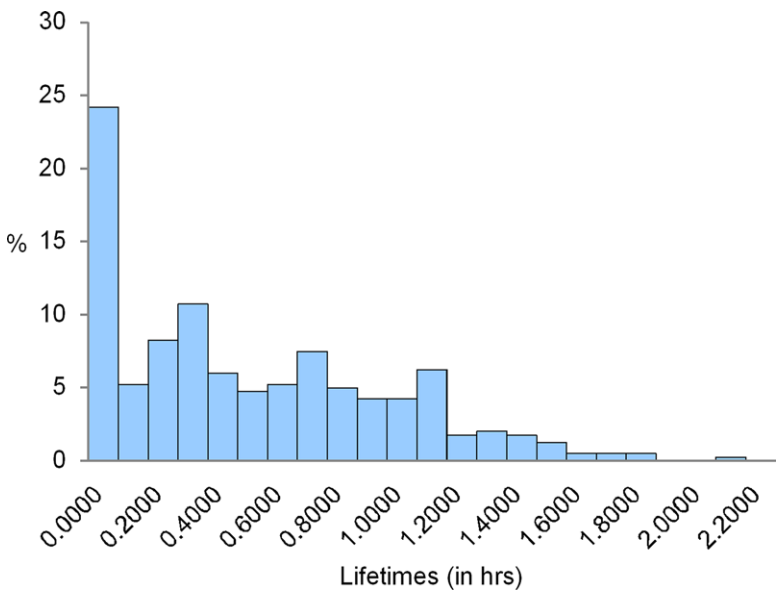

Fig. 2. The "lifetime" distribution for all flights.

revealed (see Table 5) that the mean lifetimes are all significantly different (p-value being near 0) across all flights, thus requiring us to use appropriate procedures to estimate the lifetime parameter λ for each flight. This very clearly supports the observation in practice that the arrival profiles of passengers differ, for example, depending on the time of flight.

Table 5
Results of one-way ANOVA for means of arrival distributions

Source	Sum of squares	df	Mean squares	F	p-Value
Flights	64.0274	13	4.92519	26.15	1.00×10^{-55}
Error	203.3966	1080	0.18833		
Total	267.4240	1093			

Table 6
Arrival group size frequency distribution

	Group size				
Flight	1	2	≥ 3	Total	Average group size
F01	41	12	5	58	1.38
F02	36	36	14	86	1.77
F03	28	19	14	61	1.77
F04	28	29	12	69	1.70
F05	21	20	13	54	1.99
F06	45	21	11	77	1.65
F07	30	30	9	69	1.79
F08	49	27	19	95	1.74
F09	55	27	12	94	1.56
F10	44	24	12	80	1.60
F11	34	24	10	68	1.68
F12	35	25	12	72	1.85
F13	45	28	44	117	1.54
F14	34	41	15	90	1.68
Total	525	363	202	1090	1.69
Proportion	0.4817	0.3330	0.1853	1.00	

4.2.1. Arrival group size distribution

We now consider the arrival group size distribution that is given in Table 6. The table reveals that most of the group sizes varied from 1 to 3 with only a very small percentage of passengers arriving in groups of sizes 4 or greater. So, in Table 6, we have merged all the groups of size greater than or equal to 3 into one single group. We note that the average group size for any flight was less than 2, and on the average, about 48.17% are singles with about 33.3% checking-in in pairs. We used the chi-square test ($\chi^2 = 69.38$ with a p-value of nearly 0) that rejected the hypothesis that the group size distribution is independent of the flights.

Thus, we conclude that the group size distribution is different across all flights. This result is significant because it implies that the parameters of the lifetime distributions are significantly different across flights. We can now consider each group in the population to be an independent subpopulation with its own lifetime distribution. Specifically, we assume that the subpopulation that arrives at the counter in groups of size i has an exponential lifetime distribution with parameter λ_i. Using the group size information in Table 6, we will estimate for each flight, its grand lifetime parameter λ of its lifetime distribution in Section 4.3.

It is now natural that the arrival process in our dynamic programming model should include the random group sizes. Since our model is events-based, it is possible to incorporate this fact. However, doing so would require augmenting the state space, which in turn would lead to the curse of dimensionality. Further, it will also make the total number of events a random variable, which would further complicate the analysis. To avoid these complications and since our aim is to develop a simple-to-use model, we assume that the passengers arrive singly to check-in. This is also a common assumption used in the literature (e.g., see Baxi, 2007).

4.2.2. Arrival process

At this juncture, we highlight that our use of the arrival process as governed by a death process is a notable deviation from the traditional approach of using i.i.d. interarrival times as our model requires the distribution of "lifetimes" rather than the distribution of interarrival times. In effect, our model assumes that the interarrival times depend on the remaining population size. Thus, our dynamic program keeps track of the number that is yet to arrive and recommends optimal decisions accordingly.

4.3. Estimation of the grand lifetime parameter λ

For any flight, let the set of members of the population that will arrive in groups of size $i = 1, 2, \ldots, \ell$ be called as the ith subpopulation, where ℓ is the maximum group size and is also the number of such subpopulations. We assume that each subpopulation i has its own distinct lifetime parameter λ_i. We then use the procedure of Basawa and Rao (1980) to estimate λ_i for each group $i = 1, 2, \ldots, \ell$. As our model assumes that passengers arrive singly, we need to estimate the grand lifetime parameter λ for the whole population. To that end, let us define q_i to be the probability that an arrival is from the subpopulation of group size i so that $\lambda_i = \lambda q_i$. Since q_i is known, we now have $\hat{\lambda} = \lambda_i/q_i$, resulting in ℓ estimates for λ from which we choose the "best" estimate based on the value of the probability of all passengers arriving within T.

We propose to use the following procedure:

1. For the chosen flight, develop the group frequency distribution and calculate the relative frequency q_i for each group $i = 1, 2, \ldots, \ell$.
2. For each group i, using the procedure of Basawa and Rao (1980), determine the estimate of the group's lifetime parameter λ_i.
3. First discard those estimates corresponding to outliers (i.e., with small q_i). Then, from each group, find an estimate of the grand lifetime parameter using the formula, $\hat{\lambda} = \lambda_i/q_i$. Order these estimates in the ascending order.
4. Note that the probability of all passengers arriving at or before T is Pr{the arrival instants of all passengers $\leq T\} = (1 - e^{-\lambda T})^N$. Now, choose the smallest (most conservative) of the above estimates $\hat{\lambda}$ that satisfies the inequality $\gamma \leq (1 - e^{-\hat{\lambda} T})^N < 1$, where $\gamma \in [0.9, 1)$. We highlight that by carefully choosing γ, we actually incorporate the possibility of no-shows in the model.

We illustrate the above procedure through an example. For this purpose, we have chosen flight F04 for which $N = 131$ and (approximately) $T = 1.5$. Table 7 presents the group size distribution for this flight and also the estimates for each group using the procedure of Basawa and Rao (1980).

Table 7
Estimation of the lifetime parameter for flight F04

Group size (i)	Frequency	Relative frequency (q_i)	λ_i	λ_i/q_i	Probability $(1-e^{-\hat{\lambda}T})^N$
1	28	0.4058	2.2164	5.4617	0.9644
2	29	0.4203	2.3387	5.5645	0.9694
3	6	0.0870	3.8710	44.5161	1.0000
4	4	0.0580	5.4546	94.0909	1.0000
5	1	0.0145	—	—	—
6	1	0.0145	—	—	—
Total	69				

Table 8
Estimates of the lifetime parameter for all flights

Flight	No. of arrival instants (n)	N	T (hours)	$\hat{\lambda}$	$(1-e^{-\hat{\lambda}T})^N$
F01	54	68	1.50	7.3055	0.9988
F02	86	156	1.33	6.3547	0.9672
F03	61	121	2.33	9.7971	1.0000
F04	69	131	1.50	5.4617	0.9644
F05	54	111	1.42	7.7885	0.9982
F06	77	137	1.25	9.2216	0.9987
F07	69	145	1.42	12.5132	0.9999
F08	95	185	2.33	4.2008	0.9898
F09	94	151	1.13	10.6616	0.9991
F10	80	167	1.42	26.1153	1.0000
F11	68	131	1.33	7.8311	0.9962
F12	71	129	1.33	14.8002	1.0000
F13	117	261	1.57	6.0569	0.9805
F14	90	172	2.75	4.9901	0.9998

We note that this flight had 69 instants of arrivals for check-in but the total number of passengers checked in was 131[$= 1(28) + 2(29) + 3(6) + 4(4) + 5(1) + 6(1)$]. Using the proposed procedure the best estimate for λ is $\hat{\lambda} = 5.4617$, assuming the probability of at least one no-show to be not more than 0.0356.

Table 8 displays the estimates of the lifetime parameter obtained for all flights using the above procedure. We note that midday flights (i.e., flights F07–F10) tend to have higher lifetime parameters indicating a quicker or earlier arrival of midday passengers.

4.4. Service process and estimation of the basic service rate μ

We recall our assumption on the actual service time (Section 3) at any counter as a function of a basic service time (distributed exponentially with mean $1/\mu$, the number of open counters, and the congestion that is the number in the queue). We now describe the procedure we propose to estimate

Table 9
Some descriptive statistics on service encounter

Variable	Average	Minimum	Maximum
No. of passengers seen on arrival	5.93	0	29
No. of passengers left behind on entering service	4.96	0	34
No. of documents presented	1.93	1	11
No. of bags checked in	1.35	0	8

Table 10
ANOVA for equality of service times

Source	Sum of squares	df	Mean squares	F	p-value
Flights	172.71	13	13.29	1.61	0.0789
Error	4471.28	541	8.26		
Total	4643.99	554			

the basic service rate μ. Toward this end, we first test whether the service time distribution is the same across all flights, as this would enable us to combine all the data points collected for all flights to estimate μ. Intuitively, this should be true because Singapore's Changi Airport is unique that any flight taking off from there is to an international destination. So, the check-in procedure must be the same for all flights. Now, we explain below our method for estimating the basic service rate.

Since it is impossible to track the service experience of each and every single passenger, we decided to track the sojourn of some selected passengers. This was done by attaching a data collector to an arriving passenger until that customer left the counter. Once the data were collected for the passenger, the enumerator waited for the next arrival to track that passenger's sojourn through the system. Thus, the following data were collected for a sample of marked customers for each flight: (a) time of arrival, (b) number in the queue seen by the arrival at the counter the customer joined, (c) time entering into service, (d) the number left behind in the queue (denoted by b) at the time of entering service, (e) the number of documents (i.e., passports) presented, (f) the number of bags checked in, and (g) the time leaving the counter after service. Table 4 (which was referred to earlier) summarizes the results for the means of the queueing time and service time for all flights. The "Sample size" column in Table 4 indicates the number of marked customers who were tracked by the collector. We see that the average waiting in the queue for each flight varied from a minimum of 57 seconds (F14) to a maximum of 16 minutes 28 seconds (F02). The average service time ranged from a minimum of 59 seconds (F08) to a maximum of 2 minutes 49 seconds (F07). Table 9 provides the details about the descriptive statistics on the variables mentioned above for all the flights.

4.4.1. ANOVA for the test of equality of mean service times

Table 10 presents the results of the ANOVA performed on the service times for all flights. From the results, it is clear that the mean service time is not different across flights (at a level of significance $\alpha = 0.05$). This is expected because the service provider is the same for all the flights and the flights too are regional airlines. So, we can reasonably assume that the service performance must be the

Table 11
Regression analysis for service time (STime) where Cong (i.e., congestion) corresponds to $Q + 1$

ANOVA table Source	Sum of squares	df	Mean squares	F	p-Value	
Regression	49.3543	2	24.6772	77.33	2.28×10^{-30}	
Residual	178.0690	558	0.3191			
Total	227.4233	560				
Regression output						
					Confidence interval	
Variables	Coefficients	SE	t (df= 558)	p-Value	95% lower	95% upper
Intercept	0.4129	0.0385	10.733	1.47×10^{-24}	0.3373	0.4884
LnDocs	0.5438	0.0437	12.436	1.64×10^{-31}	0.4579	0.6296
LnCong	−0.0474	0.0226	−2.0990	0.0362	−0.0917	−0.0030

same for all the flights. This is a useful result because we can now combine the service time data for all the flights to estimate the service time parameter.

4.4.2. Regression model for service times

We now embark on estimating the parameters of the service time distribution by testing the relationship between service time and the related variables. It is common knowledge that the check-in service time depends on (a) the group size (Docs, measured by the number of passports presented), (b) the number of bags checked, (c) the security-related checks, and (d) congestion in the queue defined as one more than the number of passengers left behind by a passenger on entering service, that is, $Q + 1$. Among these four explanatory variables, we ignored the number of bags checked (i.e., (b)) and the security checks (i.e., (c)), since they are highly correlated with the group size (i.e., (a)) and multicollinearity in regression analysis results in unreliable estimates. This left us with two explanatory variables, Docs and $Q + 1$ in our regression model. Thus, we hypothesize the following nonlinear model for service time:

$$\text{STime} = A(\text{Docs})^{\alpha}(Q+1)^{\gamma}\epsilon, \tag{4}$$

with $\alpha \geq 0$ and $\gamma \leq 0$ and where STime is the actual service time, Docs is the group size that is measured by the number of passports presented at the counter, Q is the number of passengers left behind in queue when a passenger enters service and ϵ is the error term.

Now, taking logarithms on both sides of the above model, we get

$$\ln(\text{STime}) = \ln(A) + \alpha \ln(\text{Docs}) + \gamma \ln(Q+1) + \ln(\epsilon), \tag{5}$$

thus rendering it a multiple linear regression model. Performing the regression with a total of 561 observations, we find the regression line as follows:

$$\widehat{\ln(\text{STime})} = 0.4129 + 0.5438 \ln(\text{Docs}) - 0.0474 \ln(Q+1).$$

For this problem, we find $F = 77.33$ with a p-value of nearly 0 as shown in Table 11 indicating that the multiple regression model is significant. That is, the results above clearly support our claim

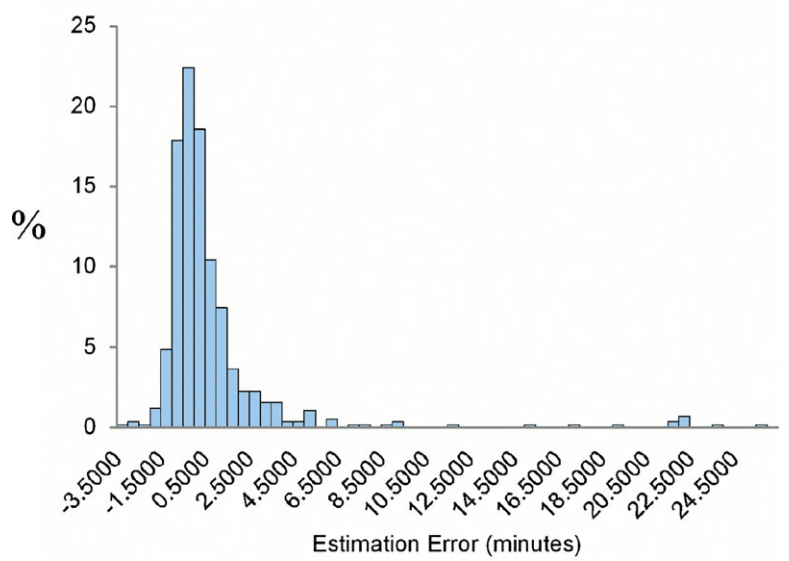

Fig. 3. Distribution of the estimation errors in the regression model.

that the service time depends on both group size and state of the queue. Returning to the original problem, we thus have

$$\widehat{\text{STime}} = e^{0.4129}(\text{Docs})^{0.5438}(Q+1)^{-0.0474},$$

from which the service time STime can be estimated for any combination of Docs and Q.

The method presented above to estimate the service time in terms of Docs and Q is quite accurate. Using the actual values of Doc and Q for all 561 observations, we calculated the estimated time $\widehat{\text{STime}}$ and compared it to the actual times. The distribution of differences (errors) between the actual and estimated times has a mean of 0.83 minutes and a standard deviation of 3.30 minutes is presented in Fig. 3. This figure shows that even though the errors range from -3.50 to 26.00 minutes, about 96% of them fall within -2.25 and 6.75 minutes, which could be considered a reasonable accuracy.

Let us now define the basic service time, BasicSTime, as the service time of a single passenger who arrives at an empty system, that is, when $Q = 0$ and Docs $= 1$. Thus, the estimate of this time is obtained as $\widehat{\text{BasicSTime}} = \widehat{\text{STime}}(\text{Docs} = 1, Q = 0) = e^{0.4129}(1)^{0.5438}(1)^{-0.0474} = e^{0.4129}$, from which the estimate of the basic service rate μ can be computed as $\widehat{\mu} = (\widehat{\text{BasicSTime}})^{-1} = A^{-1} = e^{-0.4129} = 0.6618$ per minute, or 39.71 per hour. We are now ready to present our stochastic programming formulation.

5. SDP formulation

Now that we have the exact expressions for the transition probabilities $p_{a+1}(c; a, s, k)$ and $p_{s+1}(c; a, s, k)$ in Section 3, we can optimize the system performance by deciding on whether

or not to open one more counter at each event epoch. We highlight here that optimization of the system is based on expected total cost involving the cost of making passengers wait and the cost of operating the open counters. To this end, we first develop the expected cost function during the interval between two consecutive events.

5.1. Total expected cost between two consecutive events

We define C_w as the unit cost of making a customer wait; this parameter has dimensions (\$/passenger-time). We also define C_s (\$/counter-time) as the unit variable cost, β_0 (\$/counter) as the fixed cost of opening a new counter, and β_1 (\$/counter) as the fixed penalty cost per counter for keeping idle counters open. The following (well-known) result will aid in computing the expected waiting and counter costs incurred between the occurrence of two events.

Assume that at T_n we observe that a passengers have arrived, s have been served, and there are k counters open. At this instant, we decide to open $c(= 0 \text{ or } 1)$ counters. We first calculate the expected waiting cost $W(c; a, s, k)$ that accrues until the next event time T_{n+1}.

Proposition 3. *Given the state vector (a, s, k), the expected waiting cost $W(c; a, s, k)$ that accrues between T_n and next event time T_{n+1} (for $n + 1 \leq 2N$, as we have a total of $2N$ events) is*

$$W(c; a, s, k) = \frac{C_w(a - s)}{(N - a)\lambda + [\min(a - s, k + c)]^{1+\gamma}\,(a - s)^{-\gamma}\mu}.$$

Proof. When we observe (a, s, k) at T_n, there are currently $(a - s)$ passengers in the system, who have not yet completed their check-in services. Since the system incurs a unit cost of C_w for holding each of these passengers, the expected waiting cost, given the observed values of (a, s, k) at T_n is $W(c; a, s, k) = E[C_w(a - s)(T_{n+1} - T_n) \mid T_n \text{ and } (a, s, k)]$. We note that the distribution of the difference $T_{n+1} - T_n$ is the same as the distribution of the minimum of (a) the time until the arrival of next passenger (exponential with rate $(N - a)\lambda$) and (b) the time until the departure of the next passenger in the system (also exponential with rate $[\min(a - s, k + c)]^{1+\gamma}(a - s)^{-\gamma}\mu$). Using the properties of the distribution of minimum of two exponentials, we obtain the required result. □

Note that when all arriving passengers have already been served, that is, when $a = s$, the waiting cost reduces to $W(c; s, s, k) = 0$ since there will not be anyone waiting in the queue.

The next proposition gives the expression for the expected counter cost incurred between the two epochs.

Proposition 4. *Given the state vector (a, s, k), the expected counter cost $G(c; a, s, k)$ that accrues between T_n and next event time T_{n+1} (for $n + 1 \leq 2N$) is*

$$G(c; a, s, k) = \beta_0 c + \beta_1 \max[k - (a - s), 0] + \frac{C_s(k + c)}{(N - a)\lambda + [\min(a - s, k + c)]^{1+\gamma}\,(a - s)^{-\gamma}\mu}. \quad (6)$$

Proof. Similar to the proof of Proposition 3, it is easy to see that since there will be a total of $k + c$ counters open during the period $[T_n, T_{n+1})$, the expected cost incurred for the actual operation of the $k + c$ counters is given by the last term in (6). If a decision is made to open a new counter, a fixed cost of β_0 is incurred, hence we obtain the cost term $\beta_0 c$. Finally, if the number of open counters

k exceeds the number of passengers who are already in the system resulting in idle counters, we penalize this with a fixed cost of β_1; thus obtaining the cost term $\beta_1 \max[k - (a - s), 0]$. □

Since the total cost $L(c; a, s, k)$ incurred between two successive epochs is the sum of waiting and counter costs, we have, $L(c; a, s, k) = W(c; a, s, k) + G(c; a, s, k)$. This quantity $L(c; a, s, k)$ will be incorporated into the dynamic programming functional equations that will be presented in the next section.

5.2. *Functional equations*

To solve the problem of determining whether or not to open a new counter after observing the state vector (a_n, s_n, k_n), we now formulate a stochastic dynamic program. Let the value function $V_n(a, s, k)$ be defined as the minimum expected cost to go from the occurrence of the nth event at epoch T_n when a passengers have arrived, s have been served, and there are k counters open. Observe that during the interval $[T_n, T_{n+1})$ (for $n + 1 \leq 2N$), the total expected cost of waiting plus the counter operation is given by $L(c; a, s, k)$. If the next event at T_{n+1} is an arrival, then the minimum expected cost to go is $V_{n+1}(a + 1, s, k)$, which occurs with probability $p_{a+1}(c; a, s, k)$ given in (2). Similarly, if the next event at T_{n+1} is a service completion then the minimum expected cost to go is $V_{n+1}(a, s + 1, k)$ with probability $p_{s+1}(c; a, s, k)$ given in (3). Thus, the functional equations can thus be written as

$$V_n(a, s, k) = \min_{c=0,1} \{L(c; a, s, k) + p_{a+1}(c; a, s, k) V_{n+1}(a + 1, s, k + c) + p_{s+1}(c; a, s, k) V_{n+1}(a, s + 1, k + c)\}. \tag{7}$$

The optimization problem faced by the service provider is to determine whether or not to open a new counter at a specific epoch given that the observed state vector is (a, s, k).

We observe that since a total of N passengers have been booked for a specific flight, the total number of possible events in the SDP formulation above is $M = 2N$ (as each passenger will generate two events: an arrival and a service completion). Also, it is important to note that the above functional equations are valid only for specific combinations of state and/or decision variables. Since only a total of K counters can be opened at any time, we must have that $k + c \leq K$, that is, the number of currently open counters plus the new counters cannot exceed the maximum number of available counters. Further, if the number of passengers yet to arrive is less than the number of open counters, then there is no need for additional counters. Hence, we impose the condition that if $N - a \leq k$, then we will not open any new counters, that is, $c = 0$. We now establish four groups of boundary conditions depending on the values taken by the state variables.

1. *Terminal boundary condition*: The terminal event occurs when all passengers have arrived and completed their service, that is, $a = s = N$. The system will then incur no more costs and so at the final epoch $M = 2N$ we have, $V_M(N, N, k) = 0$, for $k = 1, \ldots, K$. For a simple example with $N = 5$ passengers and a maximum of $K = 3$ available counters, the value function $V_{10}(5, 5, \mathcal{K}) = 0$ is shown in Table 12, where $\mathcal{K} = \{k : k = 1, \ldots, K\}$ and $\hat{\mathcal{K}} = \{k : k = 1, \ldots, K + 1\}$
2. *Off-diagonal boundary condition*: Diagonal states are reached when every arriving passenger has completed her service, that is, when $a = s$. From such a state, it is impossible to make

Table 12
Boundary conditions for the dynamic programming functional equation for $N = 5$ passengers and a maximum available $K = 3$ counters, where $\mathcal{K} = \{k : k = 1, \ldots, K\}$ and $\hat{\mathcal{K}} = \{k : k = 1, \ldots, K+1\}$; note that all boundary values equal ∞, except $V_{10}(5, 5, \mathcal{K}) = 0$

$a\backslash s$	0	1	2	3	4	5
0	$V_0(0,0,4)$	$V_1(0,1,\hat{\mathcal{K}})$				
1	$V_1(1,0,4)$	$V_2(1,1,4)$	$V_3(1,2,\hat{\mathcal{K}})$			
2	$V_2(2,0,4)$	$V_3(2,1,4)$	$V_4(2,2,4)$	$V_5(2,3,\hat{\mathcal{K}})$		
3	$V_3(3,0,4)$	$V_4(3,1,4)$	$V_5(3,2,4)$	$V_6(3,3,4)$	$V_7(3,4,\hat{\mathcal{K}})$	
4	$V_4(4,0,4)$	$V_5(4,1,4)$	$V_6(4,2,4)$	$V_3(4,3,4)$	$V_8(4,4,4)$	$V_9(4,5,\hat{\mathcal{K}})$
5	$V_5(5,0,4)$	$V_6(5,1,4)$	$V_7(5,2,4)$	$V_8(5,3,4)$	$V_9(5,4,4)$	$V_{10}(5,5,\mathcal{K}) = 0$
6	$V_6(6,0,\hat{\mathcal{K}})$	$V_6(6,1,\hat{\mathcal{K}})$	$V_6(6,2,\hat{\mathcal{K}})$	$V_6(6,3,\hat{\mathcal{K}})$	$V_6(6,4,\hat{\mathcal{K}})$	

a transition to an off-diagonal state where $s = a + 1$, since the next event cannot be a service completion if every arriving passenger has completed her check-in service. We assign an arbitrarily large value to the value function for such transitions. Hence, we have the following boundary conditions corresponding to these off-diagonal states $V_n(a, s, k) = \infty$, for $a = 0, \ldots, N-1$, $s = a + 1$, $n = a + s$, and $k = 1, \ldots, K + 1$. For the case with $N = 5$ and $K = 3$, these values $[V_1(0, 1, \hat{\mathcal{K}}), V_3(1, 2, \hat{\mathcal{K}}), V_5(2, 3, \hat{\mathcal{K}}), V_7(3, 4, \hat{\mathcal{K}}), V_9(4, 5, \hat{\mathcal{K}})]$ are shown in Table 12 immediately above the diagonal values.

3. *Horizontal boundary condition*: This corresponds to the case in which all passengers have arrived, that is, $a = N$, and thus a transition to $a = N + 1$ is impossible. For such transitions we assign an arbitrarily large value to the value function, that is, $V_n(a, s, k) = \infty$, for $s = 0, \ldots, N-1$, $a = N + 1$, $n = a + s$, and $k = 1, \ldots, K + 1$. For the simple case, these values $[V_6(6, 0, \hat{\mathcal{K}}), V_6(6, 1, \hat{\mathcal{K}}), V_6(6, 2, \hat{\mathcal{K}}), V_6(6, 3, \hat{\mathcal{K}}), V_6(6, 4, \hat{\mathcal{K}})]$ appear at the bottom of Table 12.
4. *Limit on the number of counters open*: For any feasible combination of (a, s) values, the total number of counters that can be opened cannot exceed the maximum available K; that is, transitions to $k = K + 1$ are impossible. For such transitions we assign an arbitrarily large value to the value function, that is, $V_n(a, s, K + 1) = \infty$, for $a = 0, \ldots, N$, $s = 0, \ldots, a$, $n = a + s$. For the case with $N = 5$ and $K = 3$, these values are shown in the middle section of Table 12.

Now that we have established the functional equations, we have to solve them to identify the optimal policy using the empirical data we have collected. The parameters to be estimated are the arrival (or death rate) parameter λ and basic service rate parameter μ. In addition, we would also want to verify statistically some of our assumptions. Toward this end, in the next section, we describe the process of data collection, the statistical analysis, and estimation.

6. Computational experiments

To demonstrate the usability of our model, we now present some numerical examples. The results shed light on the counter allocation problem and show how decisions can be made and insights gained to aid the manager. We first discuss two hypothetical examples with small values of N. The

Table 13
A five-passenger example with parameter values $(N, K, T \mid C_w, C_s \mid \beta_0, \beta_1 \mid \hat{\lambda}, \hat{\mu} \mid \gamma) = (5, 3, 1 \mid 40, 60 \mid 75, 25 \mid 5.51, 1.20 \mid -0.0474)$

n	a	s	k	c	$V_n(a, s, k)$
5	5	0	1	1	655.33
4	4	0	1	1	650.26
3	3	0	1	1	646.16
0	0	0	1	0	668.63
0	0	0	2	0	655.99
0	0	0	3	0	844.61

In this example, it is optimal to open the system with two counters resulting in a minimum expected cost of $V_0(0, 0, 2) = 655.99$.

next two examples that follow use the real arrival and service time data collected for the flights. In all cases, we use $C_w = \$40$ as the passenger delay cost and for the check-in counter operating cost we use $C_s = \$60$. We refer the readers to Parlar and Sharafali (2008) for additional details on the estimation of these costs.

Example 1. In this small-scale hypothetical example, the parameter values are $(N, K, T \mid C_w, C_s \mid \beta_0, \beta_1 \mid \hat{\lambda}, \hat{\mu} \mid \gamma) = (5, 3, 1 \mid 40, 60 \mid 75, 25 \mid 5.51, 1.20 \mid -0.0474)$. Solving the dynamic programming problem (7), we find that it is optimal to open the system with only two counters and there will never be a situation to open another counter. If for some reason, we decide to open the system with only *one* counter, then the earliest time the model recommends opening of the second counter is at T_3, if at that time three passengers have arrived and none have finished the check-in process; see Table 13, in which we display only those instances when a new counter is opened. For all other combination of (n, a, s, k) values not shown in Table 13, the optimal policy calls for not opening a new counter, that is, $c = 0$. Thus, for this case the maximum number of counters required is only 2, that is, the optimal policy is a static policy. This policy results in a minimum expected cost of \$655.99. ♦

We note that as N increases, the output will contain a large number of rows as above from which getting insights would be too difficult. So, we decided to perform some simple sensitivity analyses on a problem with a smaller N. In the next example, we perform this by taking $N = 10$ and varying μ.

Example 2. In this hypothetical example, the parameter values are chosen as $(N, K, T \mid C_w, C_s \mid \beta_0, \beta_1 \mid \hat{\lambda}, \hat{\mu} \mid \gamma) = (10, 3, 1 \mid 40, 60 \mid 75, 25 \mid 5.51, \bullet \mid -0.0474)$. Our analysis revealed that of all the parameters in the problem, μ is the one that has significant impact on the total cost. Table 14 presents the results of this analysis. Columns 2 and 3 present, respectively, the optimal number of counters to open the system with (i.e., at time $t = 0$) and the associated optimal cost. In all cases, the optimal initial number of counters to open is two. Moreover, as μ increases, cost decreases since the faster service results in shorter waiting times.

Column 4 provides information on whether we would be opening additional counters during the course of the operations and, if so, the maximum number of counters that the system will use. Column 5 gives the event that triggers the opening of the first additional counter. We note that opening the system with two initial counters, we would require an additional counter only for very

Table 14
Sensitivity of the results for varying values of the service rate parameter μ

(1) μ	(2) Optimal initial no. of counters to open (k^*)	(3) Optimal total cost ($)	(4) Max no. of counters to use	(5) Initial no. of counters $= k^*$; open next counter at event (a, s, k)	(6) Initial no. of counters $k = 1$; open next counter at event (a, s, k)
1.2	2	1418.63	3	(4,0,2)	(2,0,1)
1.5	2	1170.61	3	(4,0,2)	(2,0,1)
2.0	2	903.39	2	—	(2,0,1)
2.5	2	737.07	2	—	(2,0,1)
3.0	2	628.26	2	—	(3,0,1)
3.5	2	552.85	2	—	(3,0,1)
4.0	2	498.73	2	—	(3,0,1)
4.5	2	459.11	2	—	(3,0,1)
5.0	2	429.83	2	—	(4,0,1)
5.5	2	408.20	2	—	(4,0,1)
6.0	2	392.37	2	—	(5,0,1)
6.5	2	381.01	2	—	(7,0,1)
7.0	2	373.15	2	—	(8,0,1)
7.5	2	368.04	2	—	(9,0,1)
8.0	2	365.13	2	—	(10,0,1)
8.5	2	363.97	2	—	(10,0,1)
9.0	2	364.22	2	—	—

low values of $\mu = 1.2$ and $\mu = 1.5$ when $(a, s, k) = (4, 0, 2)$, thus increasing the total required to three.

If for some reason, we open the system with only one counter (which is sub-optimal), then column 6 indicates the event when the additional counter opening will take place. We infer that as μ increases, this happens when more and more passengers have arrived and none have cleared the check-in process.

We also note that for low values of μ, the cost is very high and we also need more counters. This implies the necessity of providing adequate training for the counter clerks to make them efficient. For high values of $\mu(\geq 2)$, the model suggests the optimal policy to be the static policy, that is, the initial number of two counters open never increases. ♦

Example 3. In this example, we consider one flight (F03, bound for Haikou, China) for which we have the actual data given in Section 4. For this flight, the input data are $(N, K, T \mid C_w, C_s \mid \beta_0, \beta_1 \mid \hat{\lambda}, \hat{\mu} \mid \gamma) = (121, 10, 2.17 \mid 40, 60 \mid 75, 25 \mid 9.7971, 39.71 \mid -0.0474)$. As the output table for this example is too large (with more than 1000 rows), we do not display the table. We observe that it is best to open with only two counters and the minimum expected cost for this would be $1751.53. As in previous examples, additional counters will be opened in the event of congestion forming due to slow service. We note that in this example, the maximum number of counters we may open during the entire operations is six. This happens only in the highly unlikely event of no one has cleared check-in and all the 121 passengers have already arrived! This should not happen in any airport unless all passengers arrive almost simultaneously. We should emphasize that even though our dynamic policy recommends starting with two counters (as was the actual case for flight F03

Table 15
Fourteen flights grouped as morning, midday, and evening flights with parameter values $(K \mid C_w, C_s \mid \beta_0, \beta_1 \mid \hat{\mu}) = (10 \mid 40, 60 \mid 75, 25 \mid 39.71)$ and the remaining parameters $(N, \hat{\lambda})$ displayed in the table

Time of day	Lifetime parameter, $\hat{\lambda}$	Flight no.	N	T (hours)	Optimal initial no. of counters to open	Optimal total cost ($)
Morning	7.3055	F01	81	1.50	2 (max = 4)	1084.66
	6.3547	F02	156	1.33	2 (max = 7)	2501.84
	9.7971	F03	121	2.33	2 (max = 6)	1751.53
	5.4617	F04	131	2.17	2 (max = 6)	2098.07
	7.7885	F05	111	1.42	2 (max = 5)	1583.70
	9.2216	F06	137	1.25	2 (max = 6)	2013.07
Midday	12.5132	F07	145	1.42	2 (max = 6)	2169.94
	4.2008	F08	185	2.33	2 (max = 8)	3295.17
	10.6616	F09	151	1.13	2 (max = 7)	2276.28
	26.1153	F10	167	1.42	2 (max = 7)	2773.51
Evening	7.8311	F11	131	1.33	2 (max = 6)	1947.57
	14.8002	F12	138	1.33	2 (max = 6)	2067.77
	6.0569	F13	261	1.57	2 (max = 10)	4669.04
	4.9901	F14	172	2.75	2 (max = 7)	2909.68

in Table 3, where the number stayed essentially constant), our policy is flexible enough to allow for possible future increases in the number of counters if that decision is justified. ♦

Example 4. We now consider all the 14 flights grouped as morning, midday, and evening flights and use the actual empirical data collected. We set $(N, K, T \mid C_w, C_s \mid \beta_0, \beta_1 \mid \hat{\lambda}, \hat{\mu} \mid \gamma) = (\bullet, 10, \bullet \mid 40, 60 \mid 75, 25 \mid \bullet, 39.71 \mid -0.0474)$ with the remaining (flight-dependent) parameters $(N, T, \hat{\lambda})$ displayed as in Table 15. We note that with the costs as assumed here, for all the flights our model recommends opening the system with an optimal number of two counters. Naturally, as the policy is dynamic, depending on the state of the system, it may be necessary to open additional counters. Table 15 also provides information on the possible maximum number of counters that we may have to open. This varies from four counters (for F01 that has only 81 passengers) to the maximum of 10 counters for F13 (with 261 as the largest number of passengers). We note again that opening of the additional counters will occur only in extreme situations of very high congestion resulting from many passengers arriving almost simultaneously. As expected, the total cost varies according to the number of passengers processed with the minimum cost of $1084.66 for flight F01 and the maximum cost of $4669.04 for F13.

Finally, we wanted to study the impact of assuming large values for fixed cost of opening an additional counter β_0 and penalty β_1 for idle counters for the case of all 14 flights. Table 16 displays the result of assuming $\beta_0 = 500$ and $\beta_1 = 50$. It is interesting to note here that when the counter setup cost β_0 is high, the model recommends a static policy for all the flights except F08 and F13, which are the flights with two largest passenger sizes (185 and 261, respectively). For the static policy, the optimal number of counters to use ranges from two (for F01) to five (for F10). ♦

We should add that we solved all the above problems, including the problems with actual data, using the computer algebra system Maple 14 on a Fujitsu Tablet PC running on Windows XP with 2 GB of RAM and a clock speed of 1.79 GHz. The maximum CPU time for all the above problems

Table 16
Fourteen flights grouped as morning, midday, and evening flights with parameter values $(K \mid C_w, C_s \mid \beta_0, \beta_1 \mid \hat{\mu}) = (10 \mid 40, 60 \mid 500, 50 \mid 39.71)$ and the remaining parameters $(N, \hat{\lambda})$ displayed in the table

Flight no.	N	Optimal policy (policy, initial no. of counters)	Optimal total cost ($)
F01	81	(Static, 2)	1391.73
F02	156	(Static, 4)	3108.50
F03	121	(Static, 4)	2271.14
F04	131	(Static, 3)	2494.17
F05	111	(Static, 3)	1985.04
F06	137	(Static, 4)	2622.19
F07	145	(Static, 4)	2924.28
F08	185	(Dynamic, 3)	4202.02
F09	151	(Static, 4)	3038.75
F10	167	(Static, 5)	3731.32
F11	131	(Static, 4)	2511.32
F12	138	(Static, 4)	2779.68
F13	261	(Dynamic, 5)	6094.55
F14	172	(Static, 4)	3668.94

was 12.281 seconds, thus showing the efficiency of our analysis in terms of the CPU time. We have also run examples for larger hypothetical flights. Fortunately, even for the largest case with $N = 700$ passengers, the CPU time was quite acceptable at approximately 4316.48 seconds. Naturally, with a faster computer the computation times would be reduced further, but for very large passenger sizes, they can easily be performed off-line. Thus, all the numerical examples demonstrate the ease with which the solution can be obtained, especially the examples that used the real data.

7. Other managerial implications

The model and analysis discussed allow managers to decide on the optimal number of counters to open and whether additional counters are required. The numerical results suggest that managers can be expected to be more confident in making better informed decisions to plan preflight allocation and make changes during the check-in process. Depending on the cost parameters, the optimal policy could be static or dynamic.

We mention that the model can be used by the manager in other ways. Most significant among these is its usability by the service provider to evaluate, *before* implementation, the performance of any counter allocation policy against standards imposed by the airport operator and the airlines. For example, a key benchmark set by the airport operator requires that no more than six passengers should be present in any queue at any time. To check whether the minimum cost allocation policy would satisfy this condition, the manager can first run the model and obtain the minimum cost allocation policy. As the underlying model is simple, one can easily program for the output to show also the number in the system at any event epoch and also flag those epochs at which the number actually exceeds the benchmark. This way, the manager can determine whether the minimum cost allocation policy meets the performance standards and also ascertain the congestion level due to

this policy. If this solution did not meet the standards, then our model can easily be programmed to open an additional counter whenever performance standards are violated. This will allow the manager to ascertain the level of congestion in the system and take steps to reduce it. The inherent flexibility in our model, which is due to the fact that our model is event based, facilitates this type of analysis.

In the airport studied, the service provider offered baggage, gate boarding, cargo, and catering services in addition to check-in services. Delays at check-in invariably impacted the timeliness of its other operations. In this respect, the model will be useful to the provider not only to meet external customer requirements, but also to align its own internal operations. We add that it can also be used in planning aspects of its operations, for example, as a manpower planning tool, since the number of counters scheduled corresponds to the number of agents planned for. Hence, for a group of flights taking off during a specified time period (e.g., in the peak period between 6 p.m. and 10 p.m.), the model can be used to determine the number of counters required for each flight and then the requirements can be time-phased to obtain an aggregate requirement plan. This aggregate requirement plan can be analyzed further to allocate counters to the airlines.

From the airlines' point of view, especially those with multiple flights departing within a given period of the day, an important decision to make is whether to use exclusive-use counters or common-use counters. The service provider on the other hand is constrained by the limited number of counters that are available. For this scenario too, the model can help managers reach decisions. All that is required is to identify flights and group them over specific intervals of time (similar to what was done in the "Arrival process" section to estimate the parameter λ). By visualizing the grouped flights as a single flight, the model can determine the overall number of counters to open during the period. In addition, it is possible to then apply the model to individual flights assuming the exclusive-use system. In effect, the two models can be compared to decide which system to use and the number of counters to allocate.

We also mention that while results directly support service performance improvements and related operational interactions, any planning tool such as the one developed here can have benefits in infrastructure planning. The airport operator has interest in capacity expansion at its terminals given the forecasted increase in passenger volumes. In particular, counter capacity estimates are required. In this respect, the model developed here can be employed as a scenario analysis tool in infrastructure planning to avoid rework resulting from inconsistent estimates as is the experience of many airports worldwide.

We should point out one criticism cited against the use of parallel queues, which is the loss of fairness, that is, a passenger who arrived early may be stuck in a slower queue compared with one who recently entered a faster moving queue.[7] This is a common observation in any parallel queue system and applies to our model too. To avoid this, one has to use only a single-queue, multiple counter system. Despite the fact that the decision maker in the model is the service provider, who may be neutral on this, if a decision is made to have a common queue (i.e., a single queue), our model can still be used. This is because our model requires only total arrivals and total departures at any event epoch and not the individual queue lengths, that is, our state space is defined in terms of total customers in the system, which is nothing but the number in the system if there were a single line only.

[7]Carl Bialik, 2009. "Justice—wait for it—on the checkout line," Numbers Guy Column. *Wall Street Journal*, 19 August 2009. Available at http://online.wsj.com/article/SB125063608198641491.html (accessed 12 June 2015) .

8. Conclusions

In this work, we put forward a new event-based dynamic programming model for the check-in counter allocation problem faced by airports around the world, especially those with an exclusive-use system. A major feature incorporated into this work is the use of real data from the award-winning Singapore Changi International Airport already known for its high service standards. The approach developed here simplifies the analysis considerably and, as demonstrated through numerical examples, is easy to use and can be applied to realistic scale problems with as many as 700+ passengers. The results provide insights that can be used to plan allocation and deploy resources dynamically. As a result, the service provider will be able to raise the standards of service to the passenger, airlines, and airport operator. This is the second work in the literature that uses dynamic optimization, which, we believe, is an improvement over other works in this area. Through numerical examples, we verify the usefulness and simplicity of the analysis, and provide managerial insights for the service provider.

Finally, we should mention about the more general problem of counter allocation to multiple airlines that have concurrent flight departures within a given period. Here, an additional factor entering the analysis is that each airline could make idiosyncratic demands (e.g., insisting that the same number of counters must be allocated as its competitor) for counters, while the number of counters available is limited. The problem becomes a constrained resource allocation problem that can be addressed through the use of constraint programming. We highlight (Parlar and Sharafali, 2008) that our model's solution will be an input to this constrained resource allocation problem.

Acknowledgments

The authors thank the Singapore Management University for their support through SMU Research Grant 07-C207-SMU-033 and the Civil Aviation Authority of Singapore for permitting us to collect the data. The authors would also like to thank the associate editor and three referees for their constructive comments that helped to improve the exposition of the paper.

References

Aczel, A., Sounderpandian, J., 2009. *Complete Business Statistics* (7th edn). McGraw-Hill/Irwin, New York.

Appelt, S., Batta, R., Lin, L., Drury, C., 2007. Simulation of passenger check-in at a medium-sized airport. Proceedings of the 2007 Winter Simulation Conference, Washington, DC, pp. 1252–1260.

Basawa, I.V., Rao, B.L.S.P., 1980. *Statistical Inference for Stochastic Processes*. Academic Press, London.

Baxi, M., 2007. Developing a model to analyze impacts of self-service and web check-in at airports. Master's thesis, School of Engineering, Cranfield University. Available at http://www.scribd.com/doc/14100315/MSc-ThesisDevleoping-a-Simulation-Model-for-Airport-CheckIn. accessed 27 January 2015).

Bhat, U.N., 1984. *Elements of Applied Stochastic Processes* (2nd edn). John Wiley, New York.

Bruno, G., Genovese, A., 2010. A mathematical model for the optimization of the airport check-in service problem. *Electronic Notes in Discrete Mathematics* 36, 703–710.

Cao, Y., Nsakanda, A.L., Pressman, I., 2003. A simulation study of the passenger check-in system at the Ottawa International Airport. Summer Computer Simulation Conference Proceedings, Montreal, Canada, pp. 573–579.

Chun, H.W., Mak, W.T.R., 1999. Intelligent resource simulation for an airport check-in counter allocation system. *IEEE Transactions on Systems, Man and Cybernetics—Part C: Applications and Reviews* 29, 3, 325–335.

Fayez, M.S., Kaylani, A., Cope, D., Rychlik, N., Mollaghasemi, M., 2008. Managing airport operations using simulation. *Journal of Simulation* 2, 41–52.
Feller, W., 1968. *An Introduction to Probability Theory and Its Applications* (3rd edn). John Wiley, New York.
Hsu, C.-I., Chao, C.-C., Shih, K.-Y., 2012. Dynamic allocation of check-in facilities and dynamic assignment of passengers at air terminals. *Computers & Industrial Engineering* 63, 410–417.
Koole, G., 1998. Structural results for the control of queueing systems using event-based dynamic programming. *Queueing Systems* 30, 323–339.
Lee, A.M., 1966. *Applied Queueing Theory*. Macmillan, London.
Minton, H., 2008. Waiting and queuing in the check-in hall: an ethnographic study of queuing and waiting for check-in services at Manchester Airport. *Journal of Airport Management* 2, 3, 249–264.
Parlar, M., Sharafali, M., 2008. Dynamic allocation of airline check-in counters: a queueing optimization approach. *Management Science* 54, 8, 1410–1424.
So, K.C., Tang, C.S., 1996. On managing operating capacity to reduce congestion in service systems. *European Journal of Operational Research* 92, 83–98.
Stolletz, R., 2010. Operational workforce planning for check-in counters at airports. *Transportation Research Part E: Logistics and Transportation Review* 46, 414–425.
van Dijk, N.M., van der Sluis, E., 2006. Check-in computation and optimization by simulation and IP in combination. *European Journal of Operational Research* 171, 1152–1168.